Autostereoscopic Display of Large-Scale Scientific Visualization

Tom Peterka[a], Robert Ross[a], Hongfeng Yu[b], Kwan-Liu Ma[c], Robert Kooima[d], and Javier Girado[e]

[a]Argonne National Laboratory, 9700 South Cass Ave., Argonne IL, USA;
[b]Sandia National Laboratories, California, PO Box 969, Livermore CA, USA;
[c]University of California at Davis, One Shields Ave., Davis CA, USA;
[d]University of Illinois at Chicago, 851 S. Morgan St., Chicago IL, USA;
[e]Qualcomm Inc., 5775 Morehouse Dr., San Diego CA, USA

ABSTRACT

Modern computational science poses two challenges for scientific visualization: managing the size of resulting datasets and extracting maximum knowledge from them. While our team attacks the first problem by implementing parallel visualization algorithms on supercomputing architectures at vast scale, we are experimenting with autostereoscopic display technology to aid scientists in the second challenge. We are building a visualization framework connecting parallel visualization algorithms running on one of the world's most powerful supercomputers with high-quality autostereo display systems. This paper is a case study of the development of an end-to-end solution that couples scalable volume rendering on thousands of supercomputer cores to the scientists' interaction with autostereo volume rendering at their desktops and larger display spaces. We discuss modifications to our volume rendering algorithm to produce perspective stereo images, their transport from supercomputer to display system, and the scientists' 3D interactions. A lightweight display client software architecture supports a variety of monoscopic and autostereoscopic display technologies through a flexible configuration framework. This case study provides a foundation that future research can build upon in order to examine how autostereo immersion in scientific data can improve understanding and perhaps enable new discoveries.

Keywords: Large scale visualization; autostereoscopic 3D displays

1. INTRODUCTION

The purpose of scientific visualization is to convert raw data into human understanding. Visualization bridges this gap, mapping bytes into forms amenable for human comprehension, and its success depends on the best management of both scientific data and human perception. The rapid growth in size and complexity of scientific data impacts both the visualization software system and the human cognitive system. Hence, as groups such as the SciDAC Institute for Ultra-Scale Visualization[1] prepare for datasets in the scale of petabytes, it is imperative to attack both problems concurrently: digital performance and human perception.

Peterka et al. are actively studying the performance problem by utilizing a new supercomputer, the IBM Blue Gene/P (BG/P), not only for computing simulations but also for visualizing the resulting data.[2] This paper links massively parallel visualization across thousands of supercomputer cores to immersive display interfaces that enhance visual understanding. In this work, we use the BG/P supercomputer to generate high-quality, illuminated stereo pairs of direct volume rendered images and stream these images to an autostereo system. The display client interleaves left- and right-eye images, and in conjunction with a parallax barrier display, presents the resulting 3D image in first-person perspective to the scientist. In this paper we discuss the modifications to the volume rendering algorithm, the transport of images from BG/P to an autostereo client, the construction of flexible client software for displaying the stream on a variety of display technologies, scientist's interaction with the display system, and overall performance of the end-to-end system. In future work, we offer suggestions to put some of these experimental ideas into practice, and ponder some of the problems that still need to be solved.

Send correspondence to Tom Peterka, E-mail: tpeterka@mcs.anl.gov

Table 1. 2008 DOE INCITE Award Winners

Scientific Domain	Data Size (TB)	Principal Investigator
Fusion	54	Klasky
Materials	100	Wolverton
Astrophysics	300	Lamb
Climate	345	Washington

Stereoscopic 3D offers perceptual advantages over monoscopic 2D. Binocular disparity is a powerful depth cue not only in computer graphics applications,[3] but in daily life as well.[4] 3D depth disambiguates data by effectively multiplying the viewable area by depth. This is especially important in scientific applications, where accurate and absolute measurement in all dimensions is a cornerstone of the display and comprehension of spatial data. Surprisingly, strict rules for plotting 2D data,[5,6] are relaxed when it comes to 3D because of limitations in viewing 3D content on 2D display devices. Depth can be approximated using indirect cues such as texture, occlusion, selective coloring[7] or global illumination.[8] These techniques improve comprehension by enhancing visual fidelity, but depth perception is still a comparative assessment. A 2D display presents indirect depth cues rather than actual depth values, but a high quality stereoscopic display affords absolute depth metrics and can offer the scientist the same level of precision in depth as in other dimensions.

Visual clutter is another problem with the 2D viewing of 3D data. As complex datasets containing millions of elements in a 3D space are projected onto a 2D plane, more than one object often maps to the same 2D location. For example, flow visualizations are prone to this problem because the tracing of a large number of particles results in many streamlines. While features such as streamlines may be well-separated in the 3D space, their 2D projections often collide, forcing the scientist to limit their number to some small amount, placing their seed points by trial and error or heuristic methods. 3D display technology can eliminate some of the guesswork from visualization and create a more complete view of data with a higher probability of showing features of interest. Figure 1 shows two examples of scientific visualizations where 3D displays can enable greater understanding.

Autostereoscopy, or the removal of stereo viewing glasses from the stereo viewing system, raises the level of engagement within the virtual world, more closely resembles a natural, human-like interface to data, and makes it possible to multiplex visualization tasks into the scientist's everyday, unpredictable workflow. Users are more likely to adopt technologies with few barriers to usage, and scientists are no exception. A scientist's day is filled with conference calls, meetings, emails, and document processing, besides interacting with data. A technology that requires the minimum impediments and merges effortlessly with other tasks is more likely to be accepted by practitioners in science fields.

Autostereo, especially at the desktop, promises to be such a technology. By combining novel display technologies with large-scale visualization executed on one of the world's most powerful supercomputers, we are researching an end-to-end visualization pipeline in response to the growing need to extract knowledge from scientific data. The coupling of highly parallel algorithms visualization algorithms and immersive interfaces is one approach to managing both the size and complexity of modern scientific data.

2. BACKGROUND

Peterka et al.[2] implemented and tested a massively parallel volume rendering algorithm on the BG/P. This is a departure from the way that such visualization is normally executed, using a dedicated GPU-accelerated visualization cluster. As scientific data grow in size, however, we must rethink the way that visualization is performed if it is to scale with the rate of growth in scientific computation. As Table 1 shows by listing the data size in terabytes of some of the largest simulations currently awarded supercomputer allocations in the U.S. Department of Energy's INCITE[9] program, scientific applications deposit hundreds of terabytes of data, and petabyte datasets are on the horizon. (The values in Table 1 are the total amount of offline data written as of June 2008. Projects are awarded for a calendar year.) A similar situation occurs in experimentally collected data as in computed simulations; increasing resolution of physical sensors translates into large data. For example, the Sloan Digital Sky Survey collects 120 megapixel images every 6 minutes, for a data accumulation rate of 8 terabytes per night.[10] In the face of growing data sizes and the costs associated with writing, reading, and

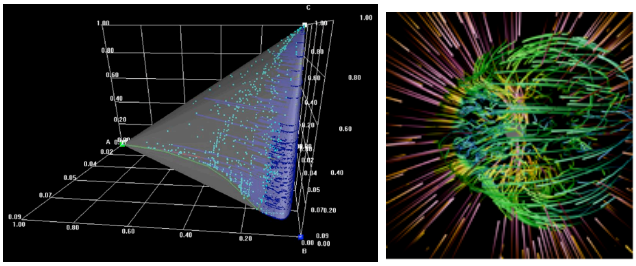

Figure 1. Two examples of visualization applications that can benefit from 3D display. Left: plotting of high-dimensional manifold surfaces. Right: vector field visualization. These applications, along with many others, are difficult to visualize when limited to a 2D view area.

Figure 2. Varrier autostereo display systems exist in a variety of form factors and sites. Left: single-tile Personal Varrier in Austin, TX. Right: 65-tile Cylindrical Varrier in San Diego, CA.

moving these data, it makes sense to bring visualization algorithms closer to the computations, rather than moving resulting data to a dedicated visualization architecture.

Hence, we are studying parallel visualization directly on the BG/P, currently ranked fifth in the Top 500 list.[11] Our volume rendering algorithm consists of three serial stages: file I/O, rendering, and compositing, each of which occurs in parallel on many cores. We demonstrated scalability up to 650 million data elements per time step on 4K BG/P cores, resulting in an end-to-end frame time of 3 seconds, including file I/O. File I/O was the largest contributing factor to the total frame time at this scale.[2] In a follow-up work, we demonstrated scalability to over one billion elements on 16K processing cores, generating high quality images up to 16 megapixels using this architecture and algorithm.[12]

However, managing performance at large scale is but one half of the modern scientific visualization problem. Datasets are three-dimensional or higher, and contain multiple variables at each spatial location. These data elements can be scalars, vectors, or tensors, and many are time-varying.[13] The added complexity can be mitigated by display systems that are capable of delivering higher information bandwidth than the customary 2D desktop display. This increased bandwidth can come in the form of higher resolution, such as tiled LCD[14] and projector walls.[15] These devices, while high resolution and wide field-of-view, are 2D and do not exploit our most effective depth discriminator: binocular vision.

Higher display bandwidth can also take the form of increased dimensionality. Stereoscopic displays began with head mounted displays (HMDs), evolved to the CAVE in 1992,[16,17] and subsequently down-sized to single-wall versions of the CAVE such as the GeoWall.[18] These technologies are still used today, and all require some form of active or passive eyewear to be worn by the user. CAVEs have also been coupled with supercomputer volume rendering by Ohno and Kageyama[19] on datasets as large as 13 GB.

Autostereo displays represent the natural next step in the evolution of virtual environments: stereo without the glasses. Quality matching that of the CAVE has been demonstrated with the Varrier system,[20] and this system has been deployed in a variety of form factors and at a number of physical sites shown in Figure 2.[21] Varrier is an example of a passive parallax barrier: two viewpoints spatially multiplexed such that only the image stripes for a given viewpoint are visible by that eye. Only two viewpoints are generated and follow the user in space through the use of a head tracking position sensing system. The parallax barrier is termed *passive* because it is a physical device mounted in front of the LCD display.

Alternatively, the parallax barrier can be *active*. In Figure 3, the printed film is replaced with a second LCD display, such that the monitor consists of a stack of two LCD layers illuminated by a common backlight.[22,23] Active barrier technology is more versatile because the barrier pattern can be changed at subpixel granularity to accommodate changes in viewer position, not to mention the ability to switch any part of the screen between 3D and 2D modes in real time. We have implemented both passive and active display types at Argonne National Laboratory.

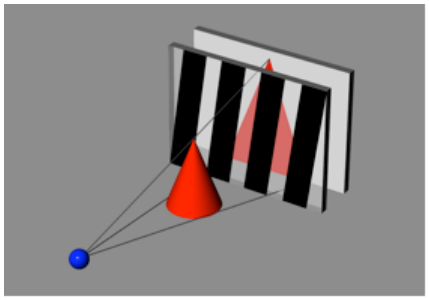

Figure 3. Parallax barrier autostereoscopy: a parallax barrier is a filter or mask placed in front of a display device. The rearmost layer above is the display device, in our case an LCD, and the layer in front of it is the parallax barrier. In a passive barrier system, the parallax barrier is a printed film or other permanent medium, while in an active barrier system, the parallax barrier is a second LCD layer that is addressable and dynamic.

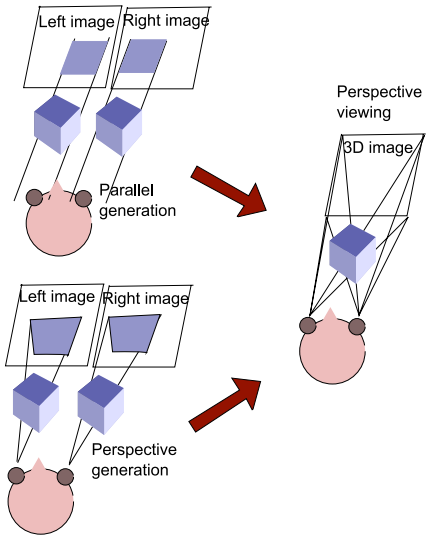

Figure 4. Top: At the supercomputer, stereo perspective projection is approximated from two orthographic projections. Bottom: Perspective projection is used in the image computation at the supercomputer instead. Right: At the display client, perspective projection is used to display the stereo image, regardless of how it was computed at the supercomputer.

3. METHOD, IMPLEMENTATION, AND RESULTS

After describing the the basic volume rendering of our test data, we discuss in detail the steps we took to convert the monoscopic volume renderer to a stereo algorithm and display the results in an autostereo environment. In particular, we describe parallel and perspective ray casting in our volume rendering algorithm, the packing and streaming of image pairs from BG/P, their remote display using a new autostereo display client library, and the interaction with the client through head tracking and a hand-held controller.

3.1 Parallel Volume Rendering of Astrophysics Data

Through our collaborations with SciDAC scientists,[24] we are visualizing astrophysics data of various physical quantities during the early stages of supernova core collapse. Variables such as pressure, density, and velocity are simulated by a computational code run by John Blondin of North Carolina State University and Anthony Mezzacappa of Oak Ridge National Laboratory.[25] Scalar data are computed on a structured grid, and in this paper we visualize entropy, stored as a 32-bit scalar value over a grid size of dimensions 864^3. Each time step is 2.5 GB. The data are time-varying, implying that each time step must be read from storage before rendering. Nearly 3/4 billion elements per time step are rendered on 4K BG/P cores. Each image is a stereo pair of left- and right-eye volume renderings of a single time step of data, read from disk, ray casted with lighting, and composited, stepping through a sequence of approximately 100 time steps on the fly.

We execute the classic software ray casting algorithm of Levoy[26] in parallel[27,28] by dividing the data statically among the 4K BG/P cores. Partial images are then composited using a sort-last scheme.[29] Ultimately the resulting image is streamed to a display device and the process repeats for the next time step. The details of parallelizing the ray casting algorithm to extremely large system and problem scales appear in.[2,12] In our current work, we modified this parallel algorithm for autostereo purposes—changing the ray casting kernel from orthographic rays to perspective rays, modifying the multiple pipeline architecture into left and right pairs, and developing a streaming mechanism to transfer the image pairs from the BG/P supercomputer to a remote display device.

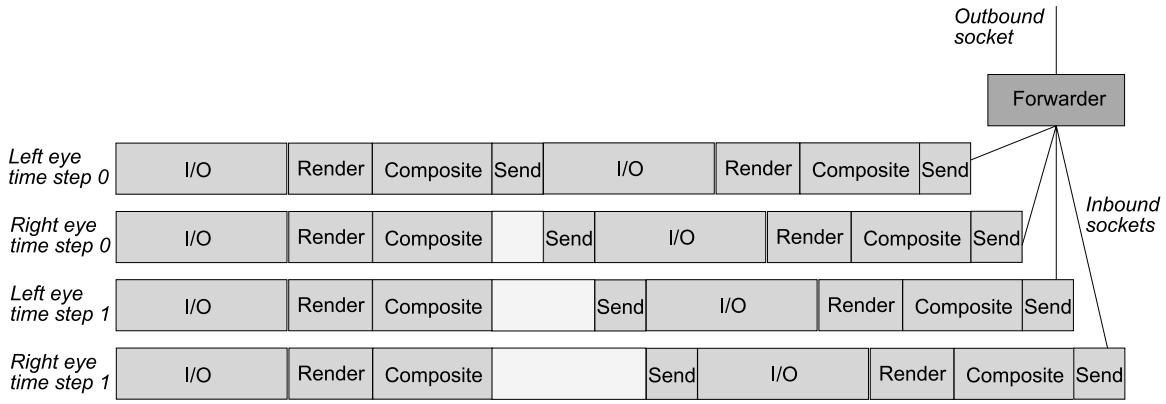

Figure 5. The total number of processor cores is partitioned into multiple pipelines. These pipes process a number of images at the same time in order to increase performance. For stereo, pipes are further grouped into left and right pairs. Each pipe sends images on a separate outbound socket to a forwarding daemon, which funnels the sockets down to one outbound link to the remote display device.

3.2 Orthographic and Perspective Ray Casting

In both human vision and photography, a lens gathers a cone of light rays, focusing them to a small region. In computer graphics, perspective projection mimics this behavior and produces familiar perspective foreshortening. Perspective projection enables depth perception through binocular disparity: a point in physical space that is imaged from different centers of projection maps to disparate positions in the corresponding images. The depth in the physical space determines the amount of disparity between corresponding points in a stereo pair of images. This is not true for orthographic projection, an alternative computer graphics technique where all viewing rays are parallel. At infinite distance, however, the two projection models are equivalent.

Originally, our volume rendering algorithm, like many others, performed ray casting using parallel (orthographic) ray directions; all rays traveled through the volume in the -Z direction. Most volume renderers are not written with stereo in mind, and parallel ray directions are easier to implement. More importantly, hierarchical data organizations are usually based on octrees, which partition the data space along orthographic planes. While our volume renderer does not utilize an octree organization, it did at one time in its history, and its ray casting method was orthographic.

Because the projection method lies deep in the kernel of the volume renderer and because we were curious to see how well stereo could be approximated with two orthographic projections, we began stereoscopic volume rendering with parallel rays. We produced two orthographic projections, shifted from each other, and displayed the result in correct perspective projection at the autostereo display. The top of Figure 4 illustrates this workflow. To display the final, combined autostereo image, the two views are spatially interleaved and textured onto a polygon in the virtual world. This polygon is set back behind the screen, as if inside the display, because orthographic projection better approximates perspective projection the farther an object is from the viewer.

The resulting polygon appeared positioned correctly in 3D, but its contents remained flat in appearance. The effect was akin to viewing a flat 2D image at some depth within a 3D space. To the casual observer, the effect of the supernova set back in space, realistically illuminated and shaded, created a plausible facsimile of 3D. However, the trained observer noticed the flatness in the center of the sphere, and we did not want to rely on 2D depth cues such as lighting and shading to convey 3D information. So, we rewrote the ray casting kernel of the volume rendering algorithm to generate perspective rays, or rays that diverge as they are traced from the center of projection through the volume. This workflow, depicted in the lower path of Figure 4, produced a deeper 3D effect when the images were textured onto a polygon at the autostereo client. The appearance of the flat polygon disappeared, and the resulting supernova took its true 3D shape instead.

3.3 Transmitting Images from Parallel Pipelines

Rather than delegating all of the available BG/P cores to volume rendering the same time step, it is more effective to group cores into *pipelines*, where each pipeline processes a different time step, concurrently with the others.

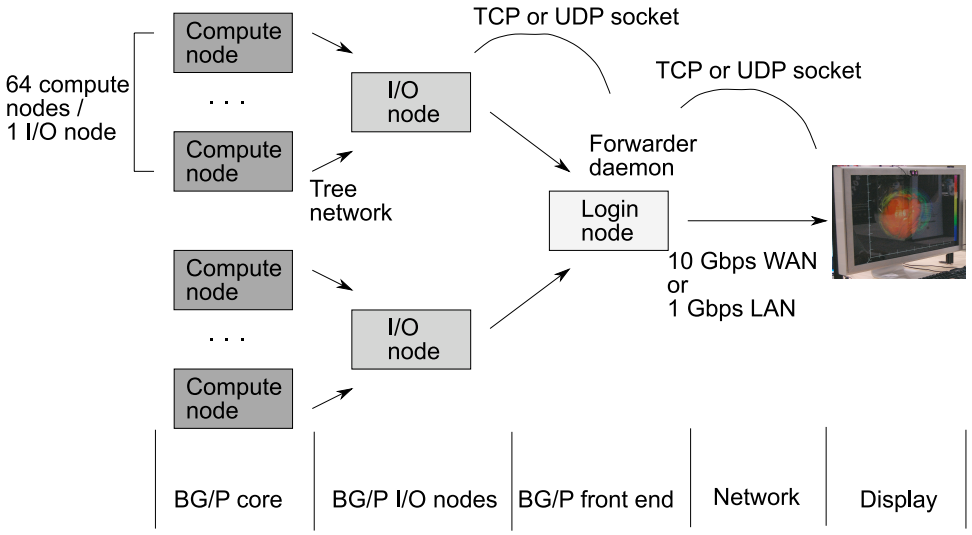

Figure 6. The data path from BG/P compute cores to the remote display device. The forwarder daemon runs on a login node. Although both TCP and UDP network protocols are available, we currently use TCP so that IP packets arrive reliably in order.

A multiple pipeline organization can hide file I/O latency.[12] To modify this algorithm for stereo, we further segmented the pipelines into two groups. Even-numbered pipes process the left eye images and odd-numbered pipes process the right eye images. Figure 5 illustrates the organization.

Images exit BG/P's compute cores via multiple, parallel sockets, one socket per pipeline. Eventually these transmissions must be serialized because the remote display client receives and displays individual stereo frames sequentially. The task of funneling multiple streams into one, properly grouping corresponding image pairs together, is performed by a forwarder daemon, also shown in Figure 5. It takes turns reading each of the incoming sockets from BG/P and forwards the header and payload sequentially on a single outbound socket to the display device. It ensures that frames are sent out in order, and metered in time so they are sent to the display at a consistent rate, despite arriving to the forwarder from the pipelines in bursts.

BG/P is organized into compute nodes, I/O nodes, and login nodes. Data transmissions require at least three physical hops to transmit information to the outside world: the first two hops are from the BG/P compute node to one of the front-end login nodes. An I/O node performs this action on behalf of the compute node to which it is assigned, so that the first hop is actually transparent to the programmer. Logically, the programmer sets up a socket from compute node to login node, followed by a socket from the login node to an outside IP address. The login node is an ideal location to run the forwarder daemon described in the previous section because it is a sequential process that terminates in a single outbound socket. Figure 6 shows the data path taken by images as they travel from compute nodes to a remote location.

Streaming applications are often written using the UDP/IP protocol because TCP/IP restricts bandwidth in order to guarantee in-order packet delivery. Our volume renderer, forwarder, and display client support both TCP and UDP network protocols, but we performed our tests with TCP, and have found it to be more useful for the time being. Our volume rendering algorithm does not generate images fast enough yet to saturate available network bandwidth. We use approximately 30 Mbps, so TCP is appropriate for now. Once our volume rendering performance or image size increases, we will require improved protocols such as Reliable Blast UDP.[30]

Logically, the left- and right-eye subimages of an image pair are combined into one image that is the original width but twice the height of a single view. This is done by setting appropriate contents of the header that precedes each subimage. Each of the individual views is sent in a separate packet with its own header, but the contents of the header reflect the fact that the two packets constitute one larger image, one subimage on top of the other. The remote display system understands the layout of the stereo image by parsing the headers of each subimage. Figure 7 shows the structure of one entire stereo image transmission.

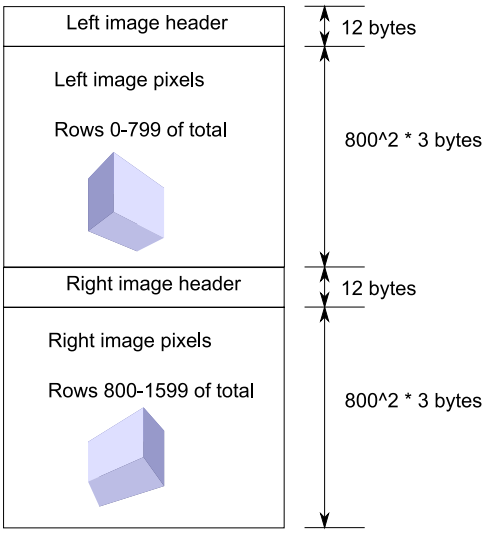

Figure 7. A sample packet structure is shown for a pair of left and right 800 x 800 images that constitute a single 800 x 1600 stereo image.

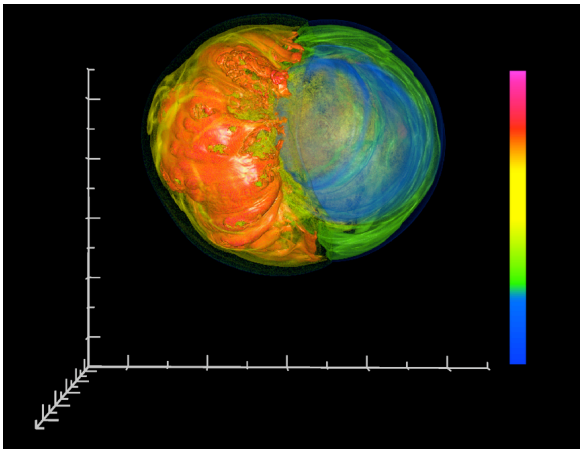

Figure 8. Dvc allows for local and remote content to be combined. Above, the supernova volume rendering was streamed from BG/P while the grid and colormap are rendered locally. The screen shot above shows in 2D what SC08 conference attendees experienced in 3D.

3.4 Remote Display Software

A client viewer displays the received images in autostereo at the scientist's workstation. This client is a small C/C++ program written using a library called *dvc*, or dynamic view client, which we wrote. Dvc supports single and multitile environments. Various options and parameters are set in a configuration file written in Lua,[31] although we are considering a more mainstream language such as Python in the future. Before dvc, we used different software platforms for passive and active barrier displays. Electro[32] is a scripting environment that supports a variety of virtual environments, including the Varrier passive barrier display,[20] but does not support active barrier displays such as the Dynallax system.[22] Dvc brings support for both passive and active barrier display technology into one library, but unlike Electro, does not have support for creating virtual worlds and scene graphs within the package.

This is intentional, because dvc is used to write lightweight clients that accept content from a variety of sources through an image streaming mechanism. In this way, software for generating scene content is divorced from the software for driving displays. This is important in scientific visualization, where software tools are well established, mature, large programs that we want to take advantage of without reinventing them. Besides acting as a live streaming client, dvc contains a flipbook utility for animating through a sequence of prerendered frames that are stored locally for later playback. Dvc supports tracking systems for input of users positional and rotational information, and navigation devices such as keyboards, mice, or 3D wands.

For a passive barrier, dvc includes the autostereo combiner method of Kooima et al.[33] to spatially combine left- and right-eye images into an interleaved autostereo image that is calibrated for the physical parameters of the parallax barrier. In the future, we intend to include support for multiview panoramagrams in dvc, enabling more than two image channels. For active barrier autostereo, dvc includes the Dynallax algorithms of Peterka et al.[23]

Besides a mechanism for accepting content from external sources, dvc is a library that the programmer can link to an openGL C or C++ program in order to augment the video stream with locally rendered content. For example, the screen shot in Figure 8 shows a supernova volume rendering being streamed from BG/P, along with grid lines and a color map that are rendered locally at the view client. This enables a combination of local and remote content, enhancing the capability and flexibility of the resulting system. Local content can originate from openGL custom rendering calls and from reading previously created models stored in the ".obj" file format.

We built three parallax barrier displays at Argonne National Laboratory for this research: two passive barrier systems and one active barrier system. They include a single-panel desktop 30-inch Personal Varrier, a six-panel

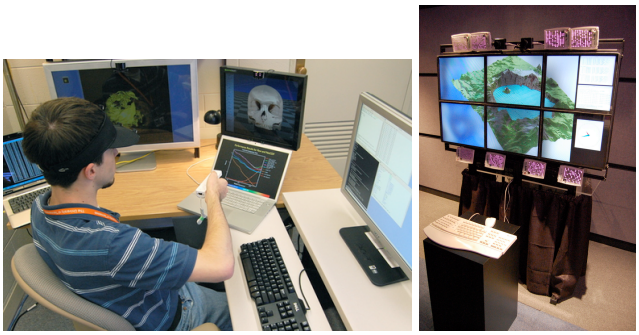

Figure 9. We deployed three autostereo systems at Argonne: a 30-inch personal Varrier display (left side of left image), a 17-inch active barrier display (center of left image), and a 60-inch 6-panel Varrier display (right image)

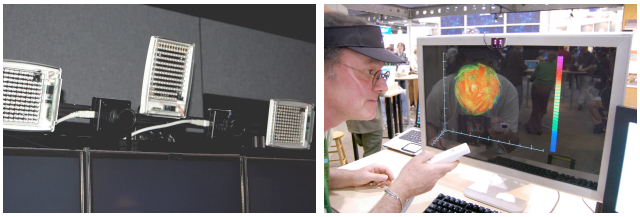

Figure 10. For the 60-inch wall display (left), we deployed a camera-based tracking system that requires no sensors to be worn. For the 30-inch desktop display (right), a low-cost game tracker is a practical solution, but requires a visor to be worn. Navigation actions such as pan, rotate, zoom, and fly are performed using a popular game controller (right).

60-inch Varrier, and an active barrier Dynallax display, all linked to the same BG/P visualizations. All three autostereo display systems at Argonne National Laboratory appear in Figure 9, and a single Linux desktop machine drives each. The machine for the 60-inch display has 4 NVIDIA 8600 GTS graphics cards. Two cards are located in PCI-Express 16X slots while two are in PCI-Express 8X slots. Since images are synchronized across all tiles, the entire graphics bus effectively runs at the slower rate of PCI-E 8X, or 2 GB/s.

3.5 Interaction

A first-person perspective, two-view autostereo display requires knowledge of the viewer's head location in order to correctly steer left- and right-eye views in space. Along with head position (x, y, z), head orientation (roll, pitch, yaw) can improve tracking accuracy. Besides maintaining correct stereo steering, head movements are interactive; this is a natural way to look around a scene when the scene is rendered from a first-person perspective that is updated in real time based on head position.

A separate tracking system sends head position and orientation to dvc. The larger 60-inch display is fitted with a custom tracking solution based on the work of Girado et al.[34] which uses a system of neural networks to process infrared video to recognize and track faces. It requires no gear to be worn and captures and computes only position information (not orientation). Knowing the head orientation as well as head position would permit greater mobility on the part of the user; but as it is, the system works well provided that the user does not rotate the head very much. We simplified the system from Girado's earlier work.[35] Just two cameras rather than three, connect to a single PC, rather than a cluster of machines as before. A dedicated Windows machine with an Intel quad-core processor executes the neural networks and sends 3D position data to dvc over a UDP socket. By averaging four camera image pixels into one before processing the image, we increased the tracking frame rate to 340 Hz. By averaging the resulting head position over several frames, we provide a more accurate, smooth result while maintaining tracking frame rate. Figure 10 (left) shows a close-up of the improved face tracking system.

The smaller desktop displays are fitted with a low-cost camera tracker designed for the home video-game industry.[36] This system tracks a triad of retroreflective spots worn on a lightweight visor. It provides full six degrees of freedom (x, y, z, roll, pitch, and yaw). We tested it in the larger wall display, but found the output to be noisy and nonlinear over the longer distance from camera to user. Figure 10 (right) shows its use in the 30-inch Personal Varrier display.

The right side of Figure 10 also shows a popular game controller that we have adopted as a hand-held, wireless navigational device. By reading the values of its accelerometers and buttons, dvc allows the user to rotate, pan, and zoom the data, as well as fly through the data space in arbitrary directions and velocities. We use the open-source cwiid library[37] and blue tooth networking to communicate between dvc and the controller.

4. PERFORMANCE

Figure 11 shows the timing results of the end-to-end process of volume rendering, streaming, and viewing stereo pairs of images from BG/P to Varrier.

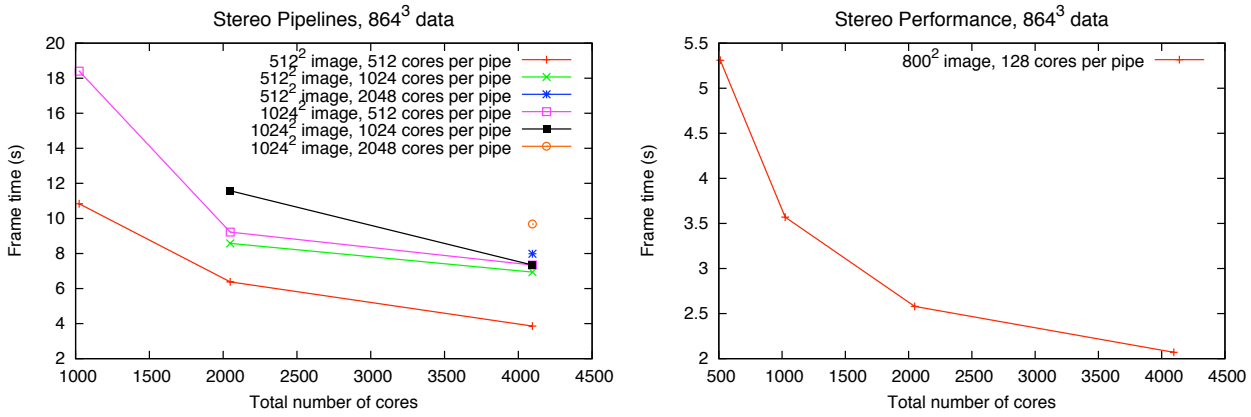

Figure 11. Left: Frame time for receiving stereo 512^2 and 1024^2 images. Performance improves when the total number of cores is divided into more pipelines (fewer cores per pipe). Right: Frame time for receiving stereo 800^2 images, from 512 to 4096 total cores, with 128 cores per pipe. 32 pipelines result in a frame time over approximately 2 seconds.

The frame time is the total interframe latency between the arrivals of new time steps at the autostereo client display. This is the time for a complete stereo image, consisting of two separate renderings of the two eye positions. The left graph shows results from tests performed in April 2008, using up to 4096 BG/P cores to generate stereo image pairs from the volume rendering algorithm described earlier. Two image sizes were tested, 512^2 and 1024^2; each of the images in the stereo pair is this size. Lighting is enabled for all tests. In those tests, the best stereo frame time for the 512^2 image was 3.9 s, and 7.3 s for the 1024^2 image.

The left graph shows that in all cases, it is more efficient to arrange the same total number of cores into a larger number of parallel pipelines, with fewer cores per pipe. In the right graph of Figure 11, we extended this concept to even higher numbers of pipes in a test of streaming 800^2 images to the SC08 conference exhibit hall. Higher numbers of pipelines together with other optimizations to the volume rendering algorithm between April and November 2008[12] resulted in a frame time of 2 seconds. 4096 cores were divided into 32 pipelines, 16 pipes for each eye.

The left image in Figure 2 and the right image in Figure 10 show the demonstration setup at the Argonne exhibit booth at SC08. There, we demonstrated the autostereo viewing of volume rendered images of supernova data. We volume rendered multiple time steps of 864^3 on BG/P on the fly at Argonne, IL, and streamed 800^2 images to Austin, TX. Over the course of a three hour period, we processed 3600 time steps, a total of 2.3 trillion data elements, or approximately 8.6 terabytes of data.

It is instructive to divide the frame time into components and see how long each part of the process takes. In the monoscopic case, we measured the time spent in the three stages of the volume rendering algorithm.[12] This is difficult to define for stereo and multiple pipelines because of the varying amounts of overlap of operations. However, we can consider the visualization step as a black box and compare it to the data transmission step. This breakout makes sense because as Figure 5 shows, the sends are nonoverlapping; we can measure their time and compare to the total frame time. The time required to send is determined by network latency and bandwidth. Increasing the pipeline parallelism changes the relative cost of visualization and transmission time because the visualization time decreases while the sending time remains constant. Table 2 shows this relationship. In the upper half of the table, a single pipeline is used for each eye, while in the lower half, the number of pipelines increases as in Figure 11.

Little is gained in the upper half of Table 2 by adding more cores because the high cost of I/O masks the rest of the process; a single pipeline for each eye does nothing to hide that cost. The fraction of transmission to total time remains 1 to 2 %. However, in the lower half, not only does the overall time improve until the I/O time is completely hidden, but the percentage of visualization time to transmission time steadily changes. In the bottom row, transmission time is nearly 15%. This is good news in terms of visualization performance, but it

Table 2. Breakdown of Total Time into Visualization and Transmission Components

Number of cores	Number of pipelines	Stereo frame time (s)	Visualization component (%)	Transmission component (%)
512	2	21.44	98.6	1.4
1024	2	12.32	97.6	2.4
2048	2	12.14	97.5	2.5
4096	2	11.66	97.4	2.6
512	4	5.31	94.4	5.6
1024	8	3.57	91.6	8.4
2048	16	2.58	88.4	11.6
4096	32	2.07	85.5	14.5

also shows that future visualization gains will need to be matched with better network performance or smarter network protocols. We will investigate these areas in our future work.

Although new time steps are generated relatively slowly by BG/P, frame refreshes at the client side were 18-24 Hz in our tests. The decoupling of generation rate and viewing rate permits the scientist to move freely, and the stereo imagery tracks the new position in real time. This is a useful feature when dealing with a slower server and network latency. However, in this scenario, user interactions are client-side only: panning and zooming the image on the screen and head movements with stereo updates of the existing image. Server-side interactions resulting in new viewpoints are currently limited by two factors. Our volume rendering algorithm is not designed for interactivity and requires longer initialization time when viewpoint changes, and the current performance would need to improve by a factor of ten in order to make complete interaction possible, modulo network latency. We will offer some suggestions to improve performance in the next section.

5. CONCLUSIONS AND FUTURE WORK

By demonstrating functionality at modest scale, we have taken the first steps toward the merging of parallel visualization with engaging work environments for scientists. Only by addressing both the scale and the perceptual issues can we successfully migrate to the petascale era of scientific computation and simulation. In this section, we offer some suggestions for improvements to our work and preview the road ahead for immersive large scale scientific visualization.

For this dataset, approximately 70% of the frame time is spent performing file I/O when rendering a single image. This is common for time-varying data, where each time step must first be read from storage. For stereo, we can cut the I/O portion of the total time in half by reorganizing the stereo pipelines so that a time step is read from disk only once for both views. We tested this reorganization and reduced frame time by another factor of 1.5.

Moreover, when combining two views into one stereo image, it is unnecessary to generate the original views in their full resolution. Approximately one half of the horizontal pixels each eye's image will be masked during the autostereo combination step with the other eye's image. This implies the original images may be generated with one half of the final horizontal resolution. For example, to generate a final 800 x 800 stereo image, each mono image can be 400 wide x 800 high. We have experimented with this idea and together with the above I/O reorganization, reduced frame time by an overall factor of 2 without noticeable loss in quality.

In the longer term, we are studying the general question of how to organize visual analysis tasks in order to maintain interactive performance at scale. This is an open topic, with both algorithmic and systemic components. For example, maintaining a local representation of a subset of the data at the client can allow rapid interaction between server updates. Image-based rendering techniques can be useful here.[38] Inserting a rendering cluster between the supercomputer and desktop is an example of a structural solution. As data grow in size, it is evident that more visualization tasks need to be performed directly at the supercomputer, but the division of labor between supercomputer, rendering cluster, and scientist's desktop is an ongoing area of research as computations move to the petascale.

Lastly, we must not forget the human in the loop. A renewed effort should focus on evaluation of interaction techniques for science, so that the end user's needs drive innovation. We computer scientists often we work from the top down, structuring systems to deliver performance criteria, without evaluating those systems from the human perspective. We admit that our own research is an example of this pattern, but we hope that it enables user evaluation in the future. As science grows not only in size but in complexity, studying human perception and how scientists engage their results becomes increasingly important.

Acknowledgments

We thank John Blondin and Anthony Mezzacappa for making their dataset available for this research. This work was supported by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357. Work is also supported in part by NSF through grants CNS-0551727, CCF-0325934, and by DOE with agreement No. DE-FC02-06ER25777.

REFERENCES

[1] [*SciDAC Institute for Ultra-Scale Visualization*] (2008). http://ultravis.ucdavis.edu/.

[2] Peterka, T., Yu, H., Ross, R., and Ma, K.-L., "Parallel volume rendering on the ibm blue gene/p," in [*Proc. Eurographics Parallel Graphics and Visualization Symposium 2008*], (2008).

[3] Ware, C. and Mitchell, P., "Reevaluating stereo and motion cues for visualizing graphs in three dimensions," in [*Proc. 2nd Symposium on Applied Perception in Graphics and Visualizaion*], 51–58 (2005).

[4] Sacks, O., "A neurologists notebook, stereo sue," *New Yorker* , 64 (June 19 2006 2006).

[5] Tufte, E. R., [*The Visual Display of Quantitative Information*], Graphics Press, Cheshire, CT, second ed. (2001).

[6] Cleveland, W. S., [*Visualizing Data*], AT&T Bell Laboratories, Murray Hill, NJ (1993).

[7] Joshi, A., Qian, X., Dione, D. P., Bulsara, K. R., Breuer, C. K., Sinusas, A. J., and Papademetris, X., "Effective visualization of complex vascular structures using a non-parametric vessel detection method," *IEEE Transactions on Visualization and Computer Graphics* **14**(6), 1603–1610 (2008).

[8] Weigle, C. and Banks, D. C., "A comparison of the perceptual benefits of linear perspective and physically-based illumination for display of dense 3d streamtubes," *IEEE Transactions on Visualization and Computer Graphics* **14**(6), 1723–1730 (2008).

[9] [*DOE Office of Science HPC*] (2008). http://hpc.science.doe.gov/.

[10] Papadomanolakis, S. and Ailamaki, A., "Workload-driven schema design for large scientific databases," *IEEE Data Engineering Bulletin* (2004).

[11] [*Top 500 Supercomputer Sites*] (2008). http://www.top500.org.

[12] Peterka, T., Ross, R., Yu, H., Ma, K.-L., Kenall, W., and Huang, J., "Assessing improvements in the parallel volume rendering pipeline at large scale," in [*Proc. SC 08 Ultrascale Visualization Workshop*], (2008).

[13] Johnson, C. and Ross, R., "Visualization and knowledge discovery: Report from the doe/ascr workshop on visual analysis and data exploration at extreme scale," tech. rep. (October 2007 2007).

[14] Leigh, J., Renambot, L., Johnson, A., Jeong, B., Jagodic, R., Schwarz, N., Svistula, D., Singh, R., Aguilera, J., Wang, X., Vishwanath, V., Lopez, B., Sandin, D., Peterka, T., Girado, J., Kooima, R., Ge, J., Long, L., Verlo, A., DeFanti, T. A., Brown, M., Cox, D., Patterson, R., Dorn, P., Wefel, P., Levy, S., Talandis, J., Reitzer, J., Prudhomme, T., Coffin, T., Davis, B., Wielinga, P., Stolk, B., Koo, G. B., Kim, J., Han, S., Kim, J. W., Corrie, B., Zimmerman, T., Boulanger, P., and Garcia, M., "The global lambda visualization facility: an international ultra-high-definition wide-area visualization collaboratory," *Future Gener. Comput. Syst.* **22**(8), 964–971 (2006).

[15] Hereld, M., Judson, I. R., Paris, J., and Stevens, R. L., "Developing tiled projection display systems," in [*Proc. International Projection Technology Workshop 2000*], (2000).

[16] Cruz-Neira, C., Sandin, D., DeFanti, T., Kenyon, R., and Hart, J., "The cave: Audio visual experience automatic virtual environment," *Communications of the ACM* **35**(6), 64–72 (1992).

[17] Cruz-Neira, C., Sandin, D., and DeFanti, T., "Surround-screen projection-based virtual reality: The design and implementation of the cave," in [*Proc. ACM SIGGRAPH 1993*], 135–142 (1993).

[18] Johnson, A., Leigh, J., Morin, P., and Van Keken, P., "Geowall: Stereoscopic visualization for geoscience research and education," *IEEE Computer Graphics and Applications* **26**(6), 10–14 (2006).

[19] Ohno, N. and Kageyama, A., "Scientific visualization of geophysical simulation data by the cave vr system with volume rendering," *Physics of the Earth and Planetary Interiors* **163**(1-4), 305–311 (2007).

[20] Sandin, D., Margolis, T., Ge, J., Girado, J., Peterka, T., and DeFanti, T., "The varrier autostereoscopic virtual reality display," *ACM Transactions on Graphics, Proceedings of ACM* **24**(3), 894–903 (2005).

[21] Peterka, T., Kooima, R., Girado, J., Ge, J., Sandin, D., and DeFanti, T., "Evolution of the varrier autostereoscopic vr display," in [*Proc. IS&T / SPIE Electronic Imaging 2007*], (2007).

[22] Peterka, T., Kooima, R., Girado, J., Ge, J., Sandin, D., Johnson, A., Leigh, J., Schulze, J., and DeFanti, T., "Dynallax: Solid state dynamic barrier autostereoscopic vr display," in [*Proc. IEEE Virtual Reality 2007*], (2007).

[23] Peterka, T., Kooima, R., Sandin, D., Johnson, A., Leigh, J., and DeFanti, T., "Advances in the dynallax solid-state dynamic parallax barrier autostereoscopic visualization display system," *IEEE Transactions on Visualization and Computer Graphics* **14**(3), 487–499 (2008).

[24] [*SciDAC*] (2008). http://www.scidac.gov/.

[25] Blondin, J. M., Mezzacappa, A., and DeMarino, C., "Stability of standing accretion shocks, with an eye toward core collapse supernovae," *The Astrophysics Journal* **584**(2), 971 (2003).

[26] Levoy, M., "Efficient ray tracing of volume data," *ACM Transactions on Graphics* **9**(3), 245–261 (1990).

[27] Ma, K.-L., Painter, J. S., Hansen, C. D., and Krogh, M. F., "A data distributed, parallel algorithm for ray-traced volume rendering," in [*Proc. 1993 Parallel Rendering Symposium*], 15–22 (1993).

[28] Yu, H., Ma, K.-L., and Welling, J., "A parallel visualization pipeline for terascale earthquake simulations," in [*Proc. Supercomputing 2004*], 49 (2004).

[29] Molnar, S., Cox, M., Ellsworth, D., and Fuchs, H., "A sorting classification of parallel rendering," *IEEE Computer Graphics and Applications* **14**(4), 23–32 (1994).

[30] He, E., Leigh, J., Yu, O., and DeFanti, T. A., "Reliable blast udp: Predictable high performance bulk data transfer," in [*CLUSTER '02: Proceedings of the IEEE International Conference on Cluster Computing*], 317, IEEE Computer Society, Washington, DC, USA (2002).

[31] Ierusalimschy, R., [*Programming in Lua*], Lua.org, second ed. (2006).

[32] Kooima, R., [*Electro*] (2008). http://www.evl.uic.edu/rlk/electro/.

[33] Kooima, R., Peterka, T., Girado, J., Ge, J., Sandin, D., and DeFanti, T., "A gpu sub-pixel algorithm for autostereoscopic virtual reality," in [*Proc. IEEE Virtual Reality 2007*], (2007).

[34] Girado, J., Sandin, D., DeFanti, T., and Wolf, L., "Real-time camera-based face detection using a modified lamstar neural network system," in [*Proc. IS&T/SPIE Electronic Imaging*], 20–24 (2003).

[35] Girado, J., Peterka, T., Kooima, R., Girado, J., Ge, J., Sandin, D., Johnson, A., Leigh, J., and DeFanti, T., "Real time neural network-based face tracker for vr displays," in [*Proc. IEEE Virtual Reality 2007*], (2007).

[36] [*TrackIR*] (2008). http://www.naturalpoint.com/trackir/02-products/product-TrackIR-4-PRO.html.

[37] [*CWiid*] (2008). http://www.wiili.org/index.php/CWiid.

[38] Zhang, C. and Chen, T., "A survey on image-based rendering–representation, sampling, and compression," *Signal Processing: Image Communication* **19**(1), 1–28 (2004).